Web Application to Identify Diseases in Tomato Leaves using Machine Learning and CNN

Oboni Anower
Birkbeck, University of London
MSc Advanced Computing

May 2024

Contents

1. Introduction

For developing countries like Bangladesh where the economy is significantly dependent on its agriculture, plant diseases can lead to deterioration in crop yield and cause a disruption in local and global supply chains. This poses a major risk of food shortage.

At early stages, symptoms of plant diseases may not be obvious, because they may look insignificant and often hard to identify. Traditionally, farmers use their years of experience and knowledge to identify diseases in crops, which is vastly due to the lack of availability of agricultural specialists in the remotest part of the region. Nonetheless, it is often time consuming and require manual labour, in some cases, quite expensive.

To overcome these day-to-day challenges faced by farmers, it is an utmost necessity to develop efficient technologies that can address disease outbreaks at early stages. One of the solutions by Md.M. Islam *et al.* (2023) suggest that a user interface leveraging Convolutional Neural Network based framework can be implemented to accelerate the disease recognition mechanism.

This academic project will focus on creating a web application integrated with machine learning or deep learning algorithms that can classify the health of tomato leaves in real-time. The process will include data acquisition, pre-processing, feature extractions, and designing models. The models will then be trained using processed data. After training, a crucial step is to measure the performance accuracy of the models to determine how reliable their disease detection will be. The models will then be compared based on their classification accuracy and generalisation ability.

This project proposal will discuss about the aim and objectives behind this project, with detailed research on previous contributions and case studies in related fields. There will be a clear explanation on how the proposed objectives are related to the existing works and describe how it will extend them with improvements. The goals will be explained in the methodologies section step by step to outline a clear understanding of the project functionalities. Additionally, an estimated workplan will be demonstrated to show the project's workplan timeline in a form of Gantt Chart. Finally, some possible risks related to the project will be summarized with suggestion of prevention strategies.

2. Aims and Objectives

The aim of this project is to design a user-friendly interface for plant health diagnosis. The interface is planned to be in a form of web page integrated with a disease classification model. It will enable the users to upload leaf images and from the input images, the system is expected to diagnose its health condition. The primary objective is to provide users with a reliable tool that will classify different types of leaf diseases precisely and efficiently. Besides, the system will ensure to helps farmers make informed decisions to plan disease control strategies and improve agricultural management.

To make an effective classification tool, the project will explore traditional machine learning algorithms as well as Convolutional Neural Networks (CNNs) and utilize them to identify a model that exhibits better accuracy. In other words, it will be attempted to examine the comparative effectiveness between different types of architectures to pick the best one for the final classification process. The performance assessment strategies of each model usually involve a range evaluation matrices like accuracy, loss, sensitivities etc, and their results help to identify the most effective model for the classification (Prabavathy et al., 2023).

A summary of the main objectives:

1. Conducting comprehensive research of existing literatures on leaf disease classification, Machine Learning and CNN architectures, and evaluation techniques; Proposing an improvement approach based on the research.

2. Data Preparation from reliable source and pre-processing the data before model training.

3. Using different model architectures and training them with pre-processed data: preferably one model using ML, and another using CNN.

4. Testing the trained models on unseen data; evaluating model complexity and performance efficiencies. Comparing the models and choose the one with higher overall classification accuracy.

5. Designing a user-friendly web application and integrating the best model to the web application.

6. Testing the system: uploading image to the web, which will give classification result.

3. Related Work

Over the years, researchers from around the world have been working on finding effective practices for early disease control in agriculture. Thanks to the advancement of machine learning technologies, many automated detection systems have been brought into attention and several papers have been published to find the best solutions. This section will discuss about some of the literatures on plant disease detection techniques using machine learning and neural networks that may be closely related to the proposal.

In a recent study, ~~Prabavathy *et al.*~~ (2023) have proposed a strategy of plant disease detection and classification models using machine learning algorithms such as Support Vector Machine (SVM), Random Forest, K-nearest neighbours etc. The purpose was to identify a plant's disease accurately to benefit farmers and agricultural researchers make preventive strategies. The research concluded that for a leaf disease dataset, the Random Forest model classified with the highest accuracy of 91.93%. Although the proposal claimed to include a CNN model for classification, it did not explicitly mention any experimental result. However, the feature extraction from the pre-processed images had been done using a CNN architecture. It is also mentioned that the classification accuracies can be further improved by using advanced deep learning methodologies. By fine-tuning CNN architectures, trained models can reach 95% or higher classification accuracies (Hu et al., 2020).

~~Durmuş *et al.* (20~~17) created a robotic platform for disease detection of tomato plants. The system would work in real-time using mobile computer and a standard RGB camera with lightweight detection code to minimise computation and hardware cost. The approach used CNN architectures like AlexNet and SqueezeNet to take advantage of their smaller size. Although both architectures showed similar classification accuracy, SqeezeNet did slightly less because of computational constrains of the hardware.

~~Md.M. Islam *et al.* (~~2023) developed a web application that could analyse plant images to identify any disease and give treatment recommendations accordingly. The motive was to help farmers take precautionary actions at an early stage to prevent financial damage. Among the three deep-learning models that were experimented on, ResNet50 architecture outperformed and gave the highest classification accuracy of 98.98%. Thus, the most accurate model was utilised in the web-application to guarantee a smart agricultural platform.

Upon carefully reviewing several research papers, it is safe to suggest that both classic machine learning models and deep learning models can detect plant diseases. Traditional machine learning methods may often require in-depth image processing like manually resizing or denoising, which can be time-consuming and resource-extensive (~~Ahmad et al., 202~~1). In contrast, deep learning techniques identifies the most relevant features from the images without explicitly mentioning which features are more important (~~S.H. Lee et al., 2017~~), even for large dataset.

Although it is noticeable that deep learning models perform with better accuracy, many external factors can affect the situation, such as, the size of dataset, data pre-processing, the computing resources. It the dataset lacks enough data, the model can be overfit by capturing noise (~~Md.M. Islam et al. , 2~~023), thus not learning the true characteristics. This may lead to not giving accurate results in real-time scenarios. Similarly, if the computation device has size constrains, fewer training samples will be process at a single iteration, hence increasing the processing time and lowering the performance of the model (~~Durmuş et al., 2017~~).

While the discussed projects focused on either using machine learning or deep learning algorithms exclusively, this proposed project plans to make a comparative analysis between both to better understand the strength and weaknesses of each type of algorithms. The suitable architectures to build these Machine Learning and CNN models will be chosen carefully after further research and understandings. Additionally, the plan is to expand the project by building a reliable web-interface to make use of the best performing model. Therefore, the ultimate motivation is to incorporate the best working model to the web application, which will detect diseases in tomato leaves as accurately as possible.

4. Proposed Methodologies

As discussed in section 2, the motivation of this project is training a Machine learning model and an end-to-end CNN architecture to evaluate the model with better accuracy. That model will then be integrated with the user interface to give users accurate disease detection. The plan is to allow users to upload images of tomato leaves and as a result it will detect whether the leaves are healthy or infected with some sort of disease.

The methodology of this proposal is broken down into several parts including data collection, image processing, algorithm selection etc, for both ML and CNN models. *Figure 1* outlines the proposed workflow for a better visualisation of the proposal.

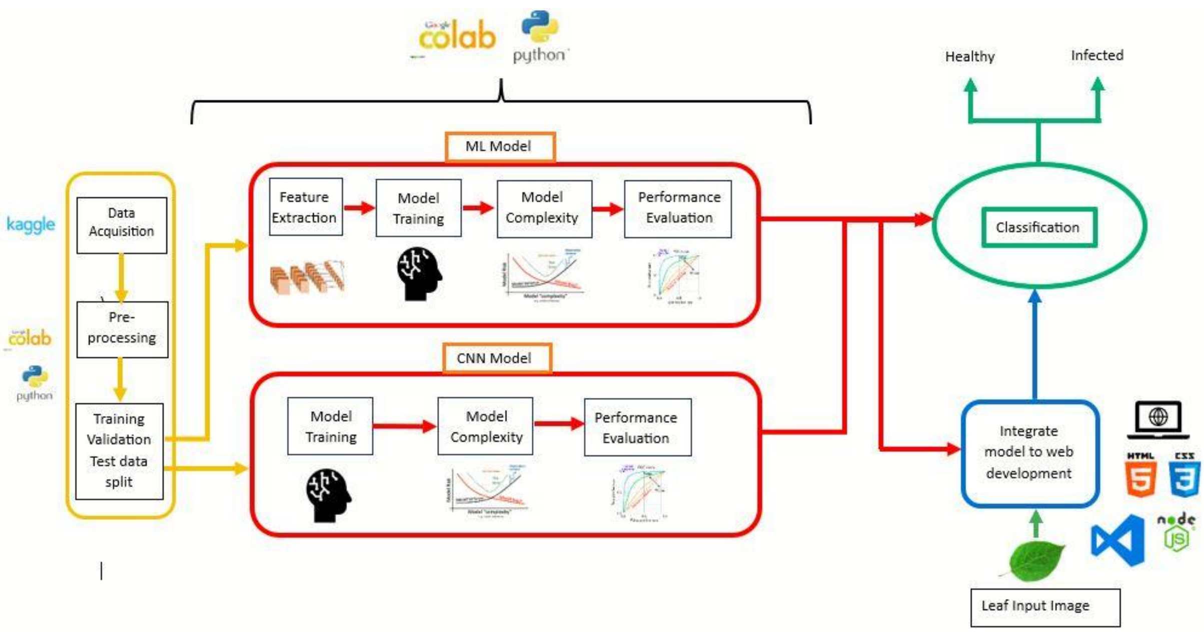

Figure 1: Workflow diagram.

4.1. Data Acquisition

The first and foremost step is to collect raw data. Generally, data are acquired either by field surveying and taking photos on camera, or collected from publicly accessible sources.

For this project, a diverse range of crop leaf images can be gathered from online source like Kaggle. The dataset being considered is "PlantVillage" which contains about 247MB of data, with around 16,012 image files (Emmanuel, 2018). These files are comprised with both healthy and diseased leaf images of different crops including tomatoes. Each type is categorised into different folders, implying that the folder name is the class label.

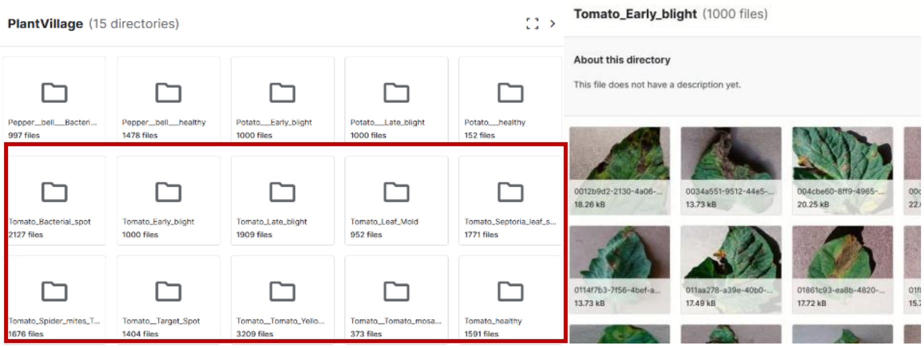

Image 1: Class Labels

Image 1 shows that there are 15 folders, indicating total 15 class labels present in the dataset. However, in this project, only 10 of the Tomato folders will be used. Each folder contains enough images related to that class. For example, class *Tomato_Early_blight* contains 1000 images of leaves that is Early blight.

This dataset is a complete with both healthy and contaminated leaves, which will be partitioned into training, validation, and testing batches. The training data will be about 75% of the dataset and be used for training the models. The remaining will be kept for evaluating the model's performance.

4.2. Data Preprocessing

Once enough images are collected, the raw data must undergo pre-processing to prevent the model from learning unwanted noise (Fulari et al., 2020). This helps to reduce model complexity and enhance the quality of the training data, while at the same time preserves (Mridha et al., 2021) the true information of data throughout the process. Common pre-processing techniques include resizing the images to a uniform size and scaling the pixel values using normalisation (Prabavathy et al., 2023).

4.3. Classification Model considerations

4.3.1. Machine learning Models

For image classification tasks, Support Vector Machine (SVM) and Random Forest (RF) are two of the top-ranked supervised machine learning algorithms by data scientists and researchers. The popularity persists because of their low computational complexity and learning ability even with small size of training data (Sheykhmousa et al., 2020). Research have shown that SVM and RF can give similar, if not better classification result as CNNs (Mboga et al., 2017).

Since a major focus in this project is assessing the comparison between traditional ML and CNN, one of these two models will be the choice of a machine learning model. The model will then be trained on the extracted features.

In ML, feature needs to be extracted manually. For this project, the **feature extraction** can be done using transfer learning. For example, a pre-trained 16-layered CNN architecture like VGG16 can be used (Prabavathy et al., 2023).

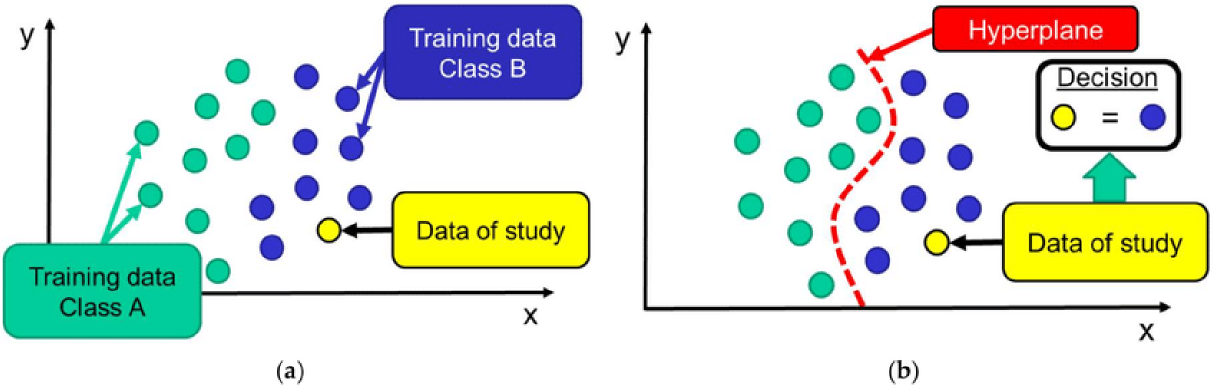

Image 2: SVM performance in 2D space (Santoyo-Ramón et al., 2018)

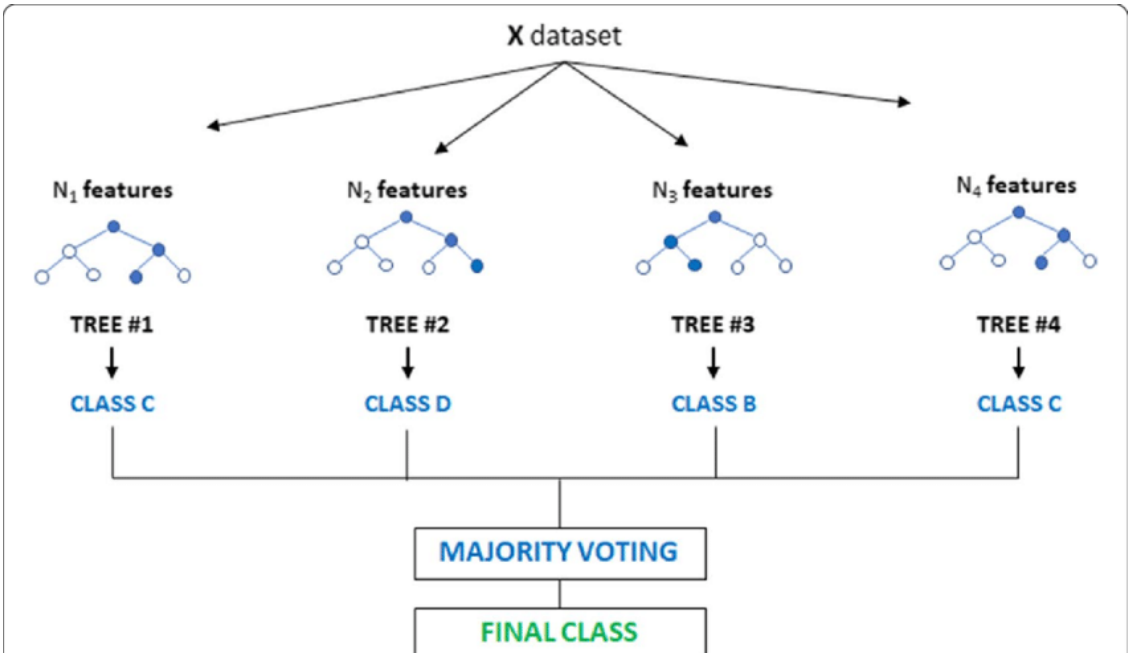

Image 3: Random Forest (Antunes , 2021)

4.3.2. CNN Architectures

CNN is a specialised deep learning architecture that is widely used for high-dimensional image data (Durmuş et al., 2017). In case of leaf disease Detection, CNN is currently the most widely used image classification technique (Lu et al., 2021). The network architectures consist of different layers, and each come with specific tasks. For example, convolutional layer handles automated feature selection (Lu et al., 2021), so a manual **feature extraction** is not required. Similarly, a pooling layer is responsible for preserving image information, and a fully connected layer handles classification task (Lu et al., 2021).

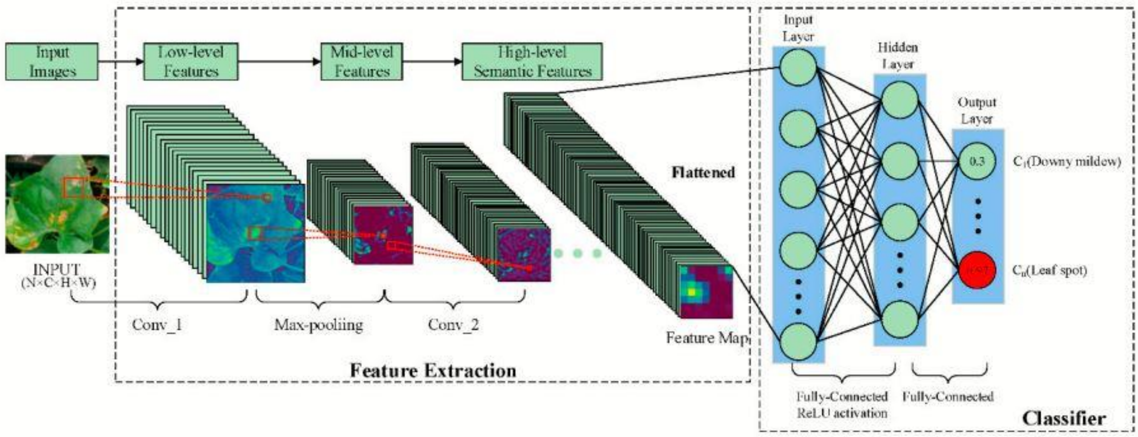

Image 4: Typical CNN for leaf disease classification (Lu et al., 2021)

As discussed in *section 3*, they offer improved accuracy compared to traditional ML and generalise better on unseen data. A number of CNN architectures have been proposed by researchers in last couple of years, such as AlexNet, ResNet, EfficientNet, VGGNet etc. AlexNet and VGG16 are simple network architectures that are commonly used for plant disease classification tasks (Lu et al., 2021). While AlexNet has slightly mmore layers than the earliest model LeNet architecture (Tang et al., 2022), VGG16 has more than AlexNet (Lu et al., 2021). Although increasing layers help the model to learn more complex patterns, aimless increasing does not aid in better classification accuracy (Lu et al., 2021) but can lead to higher training errors (Dhaka et al., 2021). However, architectures like ResNet and EfficientNet are much deeper layered networks that managed to solved the many-layer issues. ResNet tries to maintain accuracy by preventing data to overfit in extremely deep layers (Tang et al., 2022).

As mentioned earlier, AlexNet or VGG16 can be a good consideration for plant disease classification. That being said, a multi layered feed forward neural network like a sequential model can also be used for a multi-class classification task, such as plant disease classification. Deep-learning framework like Keras provides Sequential API, allowing to create a customised architecture by introducing layers one by one . Although it is simple to work with this API, the constrain is it must have only single input/outputs (Pramoditha, 2022).

For this Project, it is proposed to work with AlexNet, VGG16 or a Customised model using Sequential API, and compare the performance with the ML model.

4.4. Model Evaluation

Once the models are trained using training data, the trained model complexity can be justified using the validation dataset. During the validation phase, any hyperparameter tuning can be done using cross validation to check if the model performs well on unseen data. This will show if the models have learned the actual underlying patterns, or if they learned the training data too well by capturing noise, thus perform poorly on validation data.

Next, the test data will be used to analyse each model's performance by computing a confusion matrix (Md.M. Islam et al. , 2023). The matrix will summarise accuracy measure, model precision, F1 score and recall in a form of classification report. These reports will help to determine the better performing model.

4.5. Software and Resources

Classification Model	
Platform	Coogle Collab
Programming Language	Python
Libraries	TensorFlow/PyTorch
Computational Resource	T4 GPU (CNN) ; default CPU (ML)

Table 1: Resource to make classification models.

The main development platform for model building is chosen to be Google Collab, a cloud-based environment, because of its free availability of computational resources like GPUs and TPUs. A well serving GPU is crucial to accelerate the training process of a resource extensive task such as CNNs.

Web Interface		
Aspect	**Tools/Software/Library**	**Description**
IDE	Visual Studio Code	-
Front-end Development	HTML, CSS, JavaScript	For creatingstructure, styling, and interactivity of the web page.
Web Server	npm http-server/Python http.server,	To serve HTML, CSS, and JavaScript files locally.
Model Export	TensorFlow.js	To convert the trained model in Google Collab to suitable format for web deployment.
Web Framework or Libraries	TensorFlow.js/React /Express.js/Flask	frameworks/libraries to handle server-side logic.
Image Upload Component	HTML input element/ Third-party libraries	To allow users upload images to the web page.
Inference	TensorFlow.js	Help JavaScript to load the model and perform inference on uploaded images.

Table 2: Resources to build the user interface.

5. Project Workplan

The project has been divided into several sections such as Kernel, Proposal, Classification Model Development Phase, Web-development phase, testing, and Final Report writing. The model development phase is planned to start no later than the last week of May. That makes it about **17 weeks** of work commitment towards the project until the submission date. The timeframe of each task is an estimation, by considering the worst-case scenarios. In practice, the actual timeframe may be much shorter, or ever longer due to trial and errors.

The remaining milestones towards project completion are:

1. **Model development:** That includes data collection which is almost completed by the time of proposal finalisation. Next, the raw images will be pre-processed, and pre-processed data will be trained using different models. The model performance will be evaluated to choose the best one to move to the web-development phase. This phase is intended to complete in ***8 weeks***. The timeframe is lengthy because the workload will overlap with term 3.

2. **Web Development:** Once the classification model is finalised, it will be integrated into the web interface and the system will be tested using input images. The approximated time duration is estimated to be about ***2 weeks***.

3. **Final Report:** The report is planned to be written throughout the development phase, to stay up to date with research, development progress, and updates. The estimated time duration is set till the submission date, which is roughly about ***4 months***. However, most of the writing is expected to start after August.

The Gantt Chart shown in *Figure 2* visualises the workplan designed for this proposal.

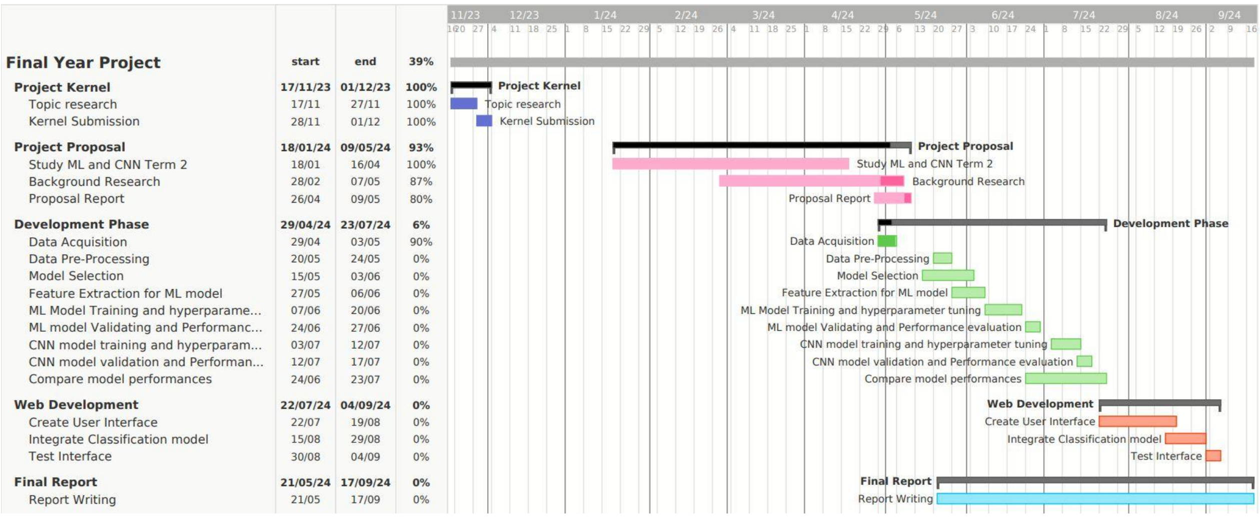

Figure 2: Work Plan for the Final Project updated on 6th May 2024

6. Ethics, Risks and Mitigation Strategies

6.1. Ethics

6.1.1. Data

Data permissions are carefully observed while collecting data to ensure compliance with regulations. For this project, data is collected from open-source data libraries like Kaggle with the recommendation from project kernel. It is made sure that the dataset does not contain personal information of any individual. The dataset may contain some inconsistency in class labelling, which is planned to be reviewed and fixed for the final project. The dataset does not acknowledge anywhere (including data source) that it was retrieved ***unethically***, by harming humans, animals, or plants.

6.1.2. Model Reliability

For the final submission, the classification models and web page will be created solely for academic purposes, and will not be considered reliable to use in field. This is to prevent any misjudgement in real-time scenarios. The models will require further assessment and licensing to be used in any real-life circumstances.

Reliability and consequences of the classification result will be disclosed to the users in the web page.

6.1.3. Transparency

Model functionalities, assumptions, limitations, and biases will be clearly explained at every stage. Any material and contributions sourced from third party will be cited appropriately in text and referenced according to academic guidelines. This includes research materials, datasets, codes, images, and network architectures.

6.2. Risks and Mitigation Strategies

Risk	**Impact Score (Out of 5)**	**Mitigation Strategy**
Misclassification of Leaf Diseases	**5**	Improve model accuracy by training on diverse datasets; Include report for confidence levels and potential errors (Deepchecks, 2022).
Model Bias and Fairness	**4**	Conduct bias assessments training-testing data to check vias-variance tradeoff (Singh, 2018).
Web Page Unavailability	**3**	Add error handling and recovery mechanisms in the application code; test the application to monitor performance regularly.
Model Overfitting	**2**	Use ensemble methods and regularization techniques to improve model's generalization (AWS, 2024).
Performance Degradation over Time	**1**	Maintain model regularly; Upgrade model with new data; Implement user feedback to continuously review performance.

Table 3: Risk Assessment

References

Ahmad et al. (2021). Plant Disease Detection in Imbalanced Datasets. *IEEE Access, 9*. Retrieved from https://ieeexplore.ieee.org/abstract/document/9568965

Antunes . (2021). To supervise or□to□self-supervise: a□machine learning based comparison on□credit supervision. 9. doi:10.1186/s40854-021-00242-4

AWS. (2024). *What is Overfitting?* Retrieved from Amazon Web Services: https://aws.amazon.com/what-is/overfitting/

Deepchecks. (2022). *Model Confidence and How it Helps Model Validation.* Retrieved from Deepchecks Community Blog: https://deepchecks.com/model-confidence-and-how-it-helps-model-validation/

Dhaka et al. (2021). A Survey of Deep Convolutional Neural Networks Applied for Prediction of Plant Leaf Diseases. *Sensors, Volume 21, Issue 14*. Retrieved from https://www.mdpi.com/1424-8220/21/14/4749

Durmuş et al. (2017). Disease detection on the leaves of the tomato plants by using deep learning. *2017 6th International Conference on Agro-Geoinformatics.* Istanbul: IEEE. Retrieved from https://ieeexplore.ieee.org/abstract/document/8047016?casa_token=jlYr-Vy33JAAAAAA:heu2aeNF5_C74jzstA88QG7Vv6cEePHxqqEZCtIGkmdfZAeo2Cg5BWZs5Lp2jB_ZTFeGGxDBTQHa

Emmanuel, T. O. (2018). PlantVillage Dataset. Retrieved from https://www.kaggle.com/datasets/emmarex/plantdisease

Fulari et al. (2020). Leaf Disease Detection Using Machine Learning. *Journal of Seybold Report*. Retrieved from https://www.researchgate.net/profile/Rajveer-Shastri/publication/344282301_Leaf_Disease_Detection_Using_Machine_Learning/links/5f6318c292851c14bc818858/Leaf-Disease-Detection-Using-Machine-Learning.pdf

Hu et al. (2020). The identification of corn leaf diseases based on transfer learning and data augmentation. *CSSE '20: Proceedings of the 3rd International Conference on Computer Science and Software Engineering.* New York: Association for Computing Machinery. doi:10.1145/3403746

Lu et al. (2021). Review on Convolutional Neural Network (CNN) Applied to Plant Leaf Disease Classification. *Agriculture, Volume 11, Issue 8*. Retrieved from https://www.mdpi.com/2077-0472/11/8/707

Mboga et al. (2017). Detection of Informal Settlements from VHR Images Using Convolutional Neural Networks. *Remote Sensing, Volume 9, Issue 11.*

Md.M. Islam et al. . (2023). DeepCrop: Deep learning-based crop disease prediction with. *Journal of Agriculture and Food Research*. Retrieved from ScienceDirect: https://www.sciencedirect.com/science/article/pii/S2666154323002715

Mridha et al. (2021). Plant Disease Detection using Web Application by. *6th International Conference on Computing, Communication and Automation.* Arad, Romania: Aurel Vlaicu University of Arad. Retrieved from https://ieeexplore.ieee.org/document/9666354/authors#authors

Prabavathy et al. (2023). Plant Leaf Disease Detection using Machine. Coimbatore. Retrieved from https://ieeexplore.ieee.org/document/10140367

Pramoditha, R. (2022). *Coding a Convolutional Neural Network (CNN) Using Keras Sequential API*. Retrieved from Towards Data Science: https://towardsdatascience.com/coding-a-convolutional-neural-network-cnn-using-keras-sequential-api-ec5211126875

S.H. Lee et al. (2017). How deep learning extracts and learns leaf features for plant classification. *Pattern Recognition*. Retrieved from https://www.sciencedirect.com/science/article/pii/S003132031730198X

Santoyo-Ramón et al. (2018). Analysis of a Smartphone-Based Architecture with. 10. doi:10.3390/s18041155

Sheykhmousa et al. (2020). Support Vector Machine Versus Random Forest for. *IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING, VOL. 13*. Retrieved from https://ieeexplore.ieee.org/document/9206124?denied=

Singh, S. (2018). *Understanding the Bias-Variance Tradeoff*. Retrieved from Towards Data SCience: https://towardsdatascience.com/understanding-the-bias-variance-tradeoff-165e6942b229

Tang et al. (2022). Image Classification based on CNN: Models and Modules. *2022 International Conference on Big Data, Information and Computer Network (BDICN).* Chongqing: IEEE. Retrieved from https://ieeexplore.ieee.org/document/9758549

Source of Images in Figure 1: workflow diagram:

https://ds100.org/course-notes-su23/probability_2/probability_2.html

https://www.linkedin.com/pulse/machine-learning-model-evaluation-metrics-fathima-zajel-liyakath/

https://www.google.com/url?sa=i&url=https%3A%2F%2Fdeepai.org%2Fmachine-learning-glossary-and-terms%2Ffeature-extraction&psig=AOvVaw1qV8VmlFmKLCoERFDPTgpZ&ust=1715301458235000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCKjhrdWq_4UDFQAAAAAdAAAAABAE

https://blog.noblinkyblinky.com/2017/09/25/connecting-visual-studio-code-to-github/vsc-logo/

https://medium.com/@afserali.b/10-best-practices-to-secure-your-node-js-application-in-production-74f045b8001c

https://www.google.com/url?sa=i&url=https%3A%2F%2Fcodeburst.io%2F25-html-css-tutorials-6a864f387185&psig=AOvVaw0a_yX5S56scbGZk7Nh8bBa&ust=1715302361952000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCMCUmLGt_4UDFQAAAAAdAAAAABAE

https://static8.depositphotos.com/1002932/796/i/600/depositphotos_7964682-stock-photo-green-leaf.jpg